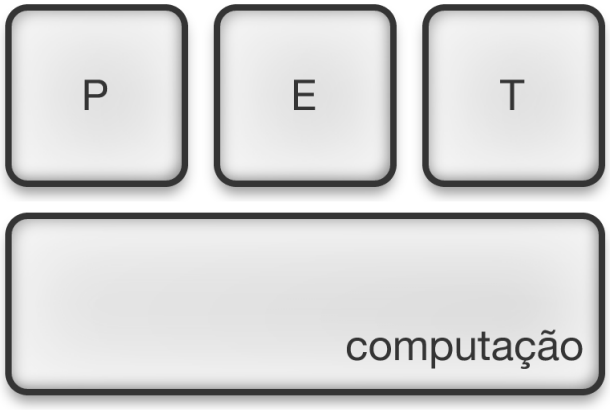

Oficina de vestibulandos

PET Computação

4 de Setembro de 2015

v0.4

Sobre este material

Este material foi escrito como material de apoio ao participantes do projeto Oficina de Vestibulandos do PET Computação da Universidade Federal do Paraná. Os objetivos são estritamente ajudar os participantes a atingir os objetivos propostos no projeto que não são necessariamente o aprendizado completo de lógica de programação ou mesmo da linguagem python, ja que ambos são apenas instrumentos necessários para uma demonstração das atividades dos cursos de computação do Departamento de Informática da UFPR.

Conteúdo

Capítulo 1

As regras da brincadeira

Para fins de simplicidade serão ensinados neste minicurso três elementos de lógica que podem ser usados para resolver problemas

1.1 Aritmética básica

As linguagens de programação permitem que você faça operações de matemática básica como em uma calculadora, são permitidas as operações: $+$, $-$, $*$ (multiplicação), / (divisão) e ainda é permitido o uso de parenteses. Por exemplo: $1 + 2$, $3 + 5 * 2$, $(3 + 5) * 2$, $5 + 4 - 2$.

1.1.1 Resto da divisão inteira

Normalmente as pessoas se esquecem do primeiro tipo de divisão ensinado nos primeiros anos do ensino fundamental, é a divisão inteira. Alguma coisa parecida com isso:

```
 894 | 35
-70  | 25
 194
-175
  19
```

Esse tipo de divisão não tem como resultado um número real, mas sim *dois* números inteiros. Um deles é o próprio resultado da divisão, no exemplo acima o número 25, que na divisão inteira representa "quantos 35's existem dentro de 894". E o segundo é o resto da divisão que representa "depois que eu tirar todos os 35's de 894 quanto sobra", no nosso exemplo 19.

1.2 Variáveis

Variaveis são nomes que damos para valores que ainda não sabemos, ou no caso de um computador valores que você só vai saber quando executar o programa.

Por exemplo na formula de bhaskara

$$ax^2 + bx + c = 0 \tag{1.1}$$

temos inicialmente 3 variáveis: a, b, e c. No meio da resolução da equacão teremos que calcular o valor de outra variável, a variavel Δ (delta). E por fim teremos os resultados que queremos que são os valores das variaveis x_1 e x_2.

Variaveis são um recurso essencial para construir programas pois a todo momento teremos que armazenar resultados em algum lugar para usa-los mais tarde.

Na computação ao contrário da matemática o nome de uma variável pode ter mais de um algarismo.

1.3 Decisão

As linguagens de programação oferecem recursos para comparar valores, e muitas vezes queremos calcular coisas diferentes dependendo do resultado dessa comparação. Usando o exemplo da formula de bhaskara, se Δ seja positivo calcularemos as raizes x_1 e x_2 da equação; se Δ for igual a 0 a equação tem apenas uma raiz então temos que calcular apenas apenas x_1; e finalmente, se Δ for negativo não existem raizes reais e portanto não devemos calcular nada, apenas mostrar uma mensagem na tela falando que é impossível realizar a operação.

Os recursos de decisão geralmente se baseiam nas comparações simples presentes na matemática. Por exemplo: $3 > 10$, $\Delta > 0$, $x = y$ (igual), $x \neq y$ (diferente).

1.4 Repetições baseadas em decisão

Existem ainda muitos casos em que queremos repetir alguns passos da solução dependendo de um critério lógico, ou seja, uma decisão como a que estudamos acima. Por exemplo, na hora de calcular o fatorial de um número n queremos realizar operação:

$$1 \times 2 \times 3 \times \ldots \times (n-2) \times (n-1) \times n \tag{1.2}$$

1. comece com o valor 1

2. enquanto não atingir n multiplique pelo próximo número

1.5 Algumas Formalidades

Seria interessante se houvessem liguagens capazes se executar comandos como o acima, no entanto a realidade é um pouco diferente devemos separar ao máximo todos os elementos de lógica que podemos usar

Errado	**Certo**
1. comece com o valor 1 **2.** enquanto não atingir n multiplique pelo próximo número	**1.** damos o valor 1 para uma variavel i **2.** damos o valor 1 para uma variavel fat **3.** enquanto $i \leq n$, faça: **3.1.** a variavel fat recebe o valor de $fat \times i$ **3.2.** incrementamos o valor da variavel i **4.** chegando aqui o valor certo está na variável fat

1.6 Exercícios

Azulejo

Calcule quantos azulejos são necessários para azulejar uma parede. É necessário conhecer a altura da parede (AP), a sua largura (LP), e a altura do azulejo (AA) e sua largura (LA).

Entrada

Para resolver o problema você receberá 4 números que representam as variáveis do enunciado. Você não sabe esse números agora e por enquanto basta saber que cada uma delas corresponde a um valor.

```
AP
LP
AA
LA
```

Considere que *AP* é multiplo de *AA* e que *LP* é multiplo de *LA*. Ou seja, não é necessário cortar um azelejo para coloca-lo na parede.

Resultado

Você deve escrever uma lógica passo-a-passo que mostre como calcular a quantidade de azulejos que serão usados para cobrir completamente a parede.

Exemplos	**Respostas**

`AP = 10`
`LP = 10`
`AA = 1`
`LA = 1`
	`100`

`AP = 6`
`LP = 4`
`AA = 2`
`LA = 1`
	`12`

`AP = 9`
`LP = 8`
`AA = 3`
`LA = 2`
	`12`

Médias

Para calcular a nota final de seus alunos, um professor de Matemática utiliza a média aritmética das notas obtidas em seis provas.

Paulo um de seus alunos ja fez quatro das seis provas e tem até agora a média 87. Paulo é caprichoso e para garantir que está se comprometendo com os estudos tem a meta de sempre atingir sua meta *ME*, mas ele ja simulou as últimas provas e a partir de seus conhecimentos atuais estimou as notas S_5 e S_6 nas provas que faltam.

Paulo quer saber duas coisas:

1. Quanto deve ser sua média nas duas provas restantes para que ele atinja sua meta *ME*
2. Qual será sua média final se ele realmente tirar as notas que previu

Entrada
O programa deve receber 3 números:
A meta que Paulo estipulou para si mesmo, e os valores previstos para as avaliações que faltam.

```
ME
S5
S6
```

Saída
Você deve mostrar a lógica para os dois problemas de Paulo.

Exemplos	**Respostas**
ME = 90 P5 = 88 P6 = 67	96 83
ME = 75 P5 = 69 P6 = 82	51 83
ME = 80 P5 = 76 P6 = 71	66 82

problema adaptado da prova específica de matemática do vestibular 2013 da UFPR

Capítulo 2

Programação e Python

Programar na realidade é escrever texto em uma linguagem textual cheia de padrões que um programa vai processar e transformar em algo que ele consegue processar. Uma dessas linguagens é o Python, uma linguagem de fácil aprendizado e bastante intuitiva. Mais informações sobre a linguagem podem ser encontradas na página da wikipedia.

2.1 O básico

Programas em linguagem python nada mais são do que vários comandos colocados em um único arquivo, eles serão executados um por um na sequência em que estão no arquivo, e ao terminar de executar cada um o computador automaticamente passará ao próximo.

Para executar seu programa python abra um terminal[1] e execute o seguinte comando:

```
python meuprograma.py
```

onde meuprograma.py é o nome do arquivo que você escreveu.

Nosso primeiro programa é uma demonstração de operações aritméticas

O que fizemos no programa acima foi usar o comando `print` (é um comando interno disponível na linguagem) para mostrar o resultado dos nossos calculos na tela do computador.

2.1.1 Variáveis

Vamos mostrar agora como computadores usam variáveis. O programa abaixo é uma boa demonstração das possibilidades do uso de variáveis.

2.2 Algumas considerações

2.2.1 comentários

Tudo aquilo em uma linha de um programa python que estiver depois de um # será ignorado pelo computador na hora de executar o programa. Esse é um recurso conhecido como *comentário* pois com ele podemos colocar no meio do programa explicações sobre como o programa funciona ou porque a lógica foi feita de tal maneira. Então os comentários não são úteis na execução do programa mas são úteis para que se entenda o código (as vezes sombrio) por trás do programa.

[1] peça ajuda a um monitor sobre isso

Programa

```
print 6+9
print 11-7
print 6*8
print 26/2
print 458+666
print 666-597
print 99*99
print 999*999
print 1+1+1
print 1+2-3+4-5
print 1+8+6-7
print 3+5*2
print (3+5)*2
print 5%2
print 17%10
print 9%3
```

Saida

```
15
4
48
13
1124
69
9801
998001
3
-1
8
13
16
1
7
0
```

Programa

```
ex = 15
print ex
print ex/3
print ex+6
soma = ex+17
print soma
soma = soma + ex + 6
print soma
print soma % 10
print soma % ex
a = 1
b = 4
c = -5
delta = b*b - 4*a*c
print delta
```

Saida

```
15
5
21
32
53
3
8
36
```

2.2.2 testando um programa com dados de entrada

Alguns programas querem que você de a ele os dados para processar ou seja eles pedem uma *entrada*. Podemos apenas digitar no teclado os dados quando o programa pedir, mas é muito mais fácil colocar todos os dados de entrada em um arquivo e mandar o programa processar este arquivo. Fazemos isso assim:

```
python meuprograma.py < dados.txt
```

No material de apoio encontrado no site você vai encontrar vários exemplos de pequenos programas para testar. Para facilitar adotamos um padrão quanto ao nome dos arquivos:

prog_x.py seu programa

prog_x.in.txt contém exemplos de entrada para o programa

prog_x.out.txt contém os resultados do programa quando ele é executado com os dados do arquivo acima. Ou seja, se o comando `python prog_x.py < prog_x.in.txt` não mostrar exatamente o conteudo de `prog_x.out.txt` o programa está incorreto.

2.3 Decisões

Na formula de bhaskara temos que seguir os seguintes passos:

1. calcular o valor de Δ, $\Delta = b^2 - 4ac$

2. se $\Delta > 0$ então

2.1 calculamos o valor de x_1,

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

2.2 calculamos o valor de x_2,

$$x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

3. se $\Delta = 0$ então

3.1 calculamos apenas o valor de x_1,

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

4. se $\Delta < 0$ então

4.1 não fazemos nada :)

Em um programa python fazemos coisa muito parecida com o esquema acima:

```
from math import sqrt

a, b, c = raw_input().split(' ')

delta = b*b - 4*a*c

if delta > 0:
    x1 = (-b + sqrt(delta))/(2*a)
    x2 = (-b - sqrt(delta))/(2*a)
    print x1, x2

if delta == 0:
    x1 = (-b + sqrt(delta))/(2*a)
    print x1

if delta < 0:
    print "nao existem raizes reais"
```

O trecho que realmente faz a comparação é o seguinte:

```
if delta > 0:
```

generalizando:

```
if condição:
```

e é isso. Agora só temos que indicar claramente ao computador quais as linhas que ele deve executar caso a comparação seja verdadeira. Fazemos isso colocando essas linhas logo abaixo do `if` e colocando um espaçamento antes dessas linhas com a tecla `tab`[2].

ou seja:

```
if <condição>:
    <linha de código pra executar>
```

2.4 Repetições

Repetições mais conhecidas como loops são um importante recurso de lógica usados na programação.

```
n = int(input())

fat = 1

while n > 1:
    fat = fat * n
    n = n-1

print fat
```

Outro caso básico é quando queremos que nossos programas executem mais de uma vez o mesmo procedimento. O programa abaixo lê e calcula o fatorial de vários números até que alguém entre com o valor 0 para que seja processado.

```
n = int(input())

while n != 0:
    fat = 1

    while n > 1:
        fat = fat * n
        n = n-1

    print fat

    n = int(input())
```

Como você deve ter percebido também não existe nenhum problema em colocar vários `while`s um dentro do outro.

[2] espaços não tem efeito nenhum, use a tecla tab

2.5 Exercícios

Azulejo

Calcule quantos azulejos são necessários para azulejar uma parede. É necessário conhecer a altura da parede (AP), a sua largura (LP), e a altura do azulejo (AA) e sua largura (LA).

Entrada

O programa deve ler 4 números (cada um em uma linha) seguindo esse formato:

```
AP
LP
AA
LA
```

Considere que *AP* é multiplo de *AA* e que *LP* é multiplo de *LA*. Ou seja, não é necessário cortar um azelejo para coloca-lo na parede.

Saída

Para as quatro linhas lidas seu programa deve mostrar apenas um número que representa a quantidade de azulejos que serão uzados para cobrir completamente a parede.

Exemplos	**Respostas**
10 10 1 1	100
6 4 2 1	12
9 8 3 2	12

Médias

Para calcular a nota final de seus alunos, um professor de Matemática utiliza a média aritmética das notas obtidas em seis provas.

Paulo um de seus alunos ja fez quatro das seis provas e tem até agora a média 87. Paulo é caprichoso e para garantir que está se comprometendo com os estudos tem a meta de sempre atingir sua meta *ME*, mas ele ja simulou as últimas provas e a partir de seus conhecimentos atuais estimou as notas P_5 e P_6 nas provas que faltam.

Paulo quer saber duas coisas:

- Quanto deve ser sua média nas duas provas restantes para que ele atinja sua meta *ME*
- Qual será sua média final se ele realmente tirar as notas que previu

Entrada

O programa deve receber 3 números:

A meta que Paulo estipulou para si mesmo, e os valores previstos para as avaliações que faltam.

```
ME
P_5
P_6
```

Saída

Exemplos	**Respostas**
90 88 67	96 83
75 69 82	51 83
80 76 71	66 82

problema adaptado da prova específica de matemática do vestibular 2013 da UFPR